PROFESSOR: I just want to be explicit about it because with all of the abstraction of QAM looking at it in terms of Hilbert filters and all of that, you tend to forget that what's really going on in terms of implementation is something very, very simple. We said that what we were doing with QAM is, we're trying to essentially build two PAM systems in parallel. So what we do is, we have two different data streams coming in. Each one there's a new signal, each t seconds. t is the time separation between times when we transmit something.

So the sequence of symbols come in. What we form is then -- well, actually this isn't a real operation. Because everybody does this in whatever way suits them. And the only thing that's important is getting from here over to there. And what we're doing in doing that is taking each of these signals, multiplying them by a pulse, to give this waveform the pulses delayed according to what signal it is. And if you like to think of that in terms of taking the signal and viewing it as an impulse, so you have a train of impulses weighted by these values here. And then filtering that, fine. That's a good way to think about it. But, at any rate you wind up with this.

With the other data stream which is going through here -- yes?

AUDIENCE: [INAUDIBLE]

PROFESSOR: You couldn't argue that was band limited also.

AUDIENCE: [INAUDIBLE]

PROFESSOR: But it's band limited in the same way. I mean, visualize -- I mean, the Fourier transform of p of t minus k t is simply the Fourier transform of p of t, which is multiplied by a sinusoid in f at this point. So it doesn't change the bandwidth of that.

In other words, when you take a function and you move it in time, the Fourier transform is going to be rotating in a different way. But its frequency components will not change at all. All of them just change in phase because of that time change.

AUDIENCE: [INAUDIBLE]

PROFESSOR: When you look at it in frequency, well no, because we're adding up all these things in frequency at that point. But each one of these pulses is still constrained within the same bandwidth.

AUDIENCE: [INAUDIBLE]

PROFESSOR: The transform of a pulse train does not look like a pulse train in frequency, no. When you take a pulse train in time, and you take the transform of each one of these pulses, what you get is a pulse which, where in frequency there is a rotation. In other words, the phase is changing. But in fact, what you wind up with is a pulse within the same frequency band. I mean, simply take the Fourier transform and look at it, and see which frequencies it has in it. In they don't change. And when you weight it with coefficients, the Fourier transform will be scaled, will be multiplied by a scalar. But that won't change which frequencies are there, either. So in fact, when you do this -- I should make that clearer in the notes. It's a little confusing.

AUDIENCE: [INAUDIBLE]

PROFESSOR: It'll be band limited in the same way, though, yes.

AUDIENCE: [INAUDIBLE]

PROFESSOR: It won't look the same. But the Nyquist criterion theorem remains the same. I mean, if you look at the proof of it, it's -- well I'll go back and look at it. I think if you look at it, you see that it actually -- I think you see that this time variation here in fact does not make any difference. I mean, I know it doesn't make any difference. But the question is, how to explain that it doesn't make a difference. And I'll go through this. And I will think about that. And I'll change the notes if need be. But anyway, it's not something you should worry about.

Anyway. What happens here is, you get this pulse train, which is just a sum of these pulses. And they all interact with each other. So at this point we have intersymbol interference. We multiply this by a cosine wave. We multiply this by a sine wave. And what we get then is some output, x of t. Which is in the frequency band. But if you take the low pass frequency here, which goes from 0 up to sum w, up at passband, we go from f c minus w to f c plus w.

Same thing here. So we have this function, which is then constrained. But it's a passband constraint to twice the frequency which we have here. When we take x sub t, and we try to go through a demodulator. The thing we're going to do is to multiply, again, by the cosine. Multiply again by the sine. Incidentally, this is minus sine and minus sine here. When you implement it, you can use plus sine or minus sine, whichever you want to. So long as you use the same thing here and there. It doesn't make any difference. It's just that when you look at things in terms of complex exponentials, this turns into a minus sine. Oh, and this turns into a minus sine.

We then filter by q of t. We filter this by q of t. The same filter. And we wind up then going through the T-spaced sampler. The T-spaced sampler and the output use of k prime. And the reason for this is that if you look at the system broken right here, this is simply a PAM waveform. And when we take this PAM waveform, the only thing we're doing is we're multiplying it by a cosine. Then we're multiplying it by a cosine again. When we get a cosine squared term, half of it comes back down to baseband. Half of it goes up to twice the carrier frequency. And this filter, in principle, is going to filter out that part at 2 times the carrier frequency. And it's also going to do the right thing to this filter. To turn it into something where, when we sample, we actually get these components back again.

So the only thing that's going on here is simply, this is a PAM system where, when we multiply by cosine twice, the thing we're getting is a baseband component again, which is exactly the same as this quantity here. Except it's -- why, I when I multiply this by this, don't I have to divide by -- don't I have to multiply by 2? I don't know. But we don't. And when we get all through the thing we come out with u sub k

prime, and we come out with u sub k double prime.

It is easier to think about this in terms of complex frequencies. When we think about it in terms of cosines and sines, we're doing it that way because we have to implement it this way. Because if you implement a Hilbert filter, you're taking a complex function, filtering it by complex waveform, which means you've got to build four separate filters. One to take the real part with the real part of the filter. The real part with the complex part of the filter. Also the imaginary part of the waveform with the real part of the filter, and so forth. So you don't implement it that way, but that's the easy way to look at what's happening. It's the easy way to analyze it. This just does the same thing.

When we use this double sideband quadrature carrier implementation of QAM, the filters p and q really have to be real. Why do they have to be real? Well, because otherwise, we're mixing what's going on here with what's going on here. In other words, you can't take -- this will not be a real waveform, so you can't just multiply it by a cosine wave. In the other implementation we had, you have this, which is a complex waveform, plus this, which is another complex waveform. You multiply it by e to the 2 pi i f c t. And then you take the real part of it. And everything comes out in the wash. Here we need this to be real and this to be real in order to be able to implement this just by multiplying by cosine and multiplying by sine.

So the standard QAM signal set makes it just parallel PAM systems. If, in fact, you don't use a standard QAM set; namely, if these two things are mixed up in some kind of circular signal set or something, then what comes out here has to be demodulated in the same way. So you don't have the total separation. But the real part of the system is the same in both cases. I think the notes for a little confusing about that.

Finally, as a practical detail here, we have talked several times about the fact that every time people do complicated filtering, and complicated filtering involves very sharp filters. Every time you make a very sharp filter at baseband, the way you do it is, you do it digitally. Namely, you first take this waveform. You sample it very, very

rapidly at a rapid pace. And then you take those digital signals and you pass those through a digital filter. And that gives you the waveform that you're interested in. When you do that, you're faced with having a waveform coming into the receiver which has a band down at baseband. Which it has width w.

You also have a band up at twice the carrier frequency. And if you sample that whole thing, according to all of the things we've done with aliasing and everything, those samples mix what's up at twice the carrier frequency with what's down here at baseband. And you really have to do some filtering to get rid of once or twice the carrier frequency.

This really isn't much of a practical issue, because you can hardly avoid doing that much filtering just in doing the sampling. I mean, you can't build the sampler that doesn't filter the waveform a little bit too. But you have to be conscious of the fact that you have to get rid of those double frequency terms before you go into the digital domain. So, you have to think of it as a little bit of filtering followed by a rapid sampler, followed by a careful filtering. Which does this function q of t here.

OK. Want to talk a little bit about the number of degrees of freedom we have when we do this. If we look at quadrature amplitude modulation, anytime you get confused about bandwidths, it's probably better to look at the sample time, t, which is something that never changes. And that always remains the same.

When you look at the sample spacing, the baseband bandwidth, as we've seen many times, the nominal baseband bandwidth is 1 over 2t. Namely, that's what's called the Nyquist bandwidth. It's what you get if you use sinc functions, which strictly puts everything in this baseband and nothing outside of it. And we've seen that by using the Nyquist criterion and building sharp filters, we can make the actual bandwidth as close to this as we want to make it.

When we modulate this up to passband, we're going to have a passband bandwidth which is 1 over t. Because, in fact, this baseband function is going from minus 1 over 2t to plus 1 over 2t. When we modulate that up, we get something that goes from f c minus 1 over 2t to f c plus 1 over 2t. Which is really an overall bandwidth of

1 over t.

If we look at this over a large time interval, t0, we have two degrees of freedom. Namely, the real part and the imaginary part. Namely, we have two different real number signals coming into this transmitter. Each t seconds. So that over a period of t0, some much larger period, we have t0 over capital T different signals that we're transmitting. Each of those signals has two degrees of freedom in it. So we wind up with 2 t0 times w real degrees of freedom. Namely, 2 t0 over T. There's 2 t0 times w, where w is the passband bandwidth.

What I want to do now, to show you that actually nothing is being lost and nothing is being gained, is to look at a baseband system where you use pulse amplitude modulation. But, in fact, you use a humongously wide bandwidth to do this. So it becomes something like these ultra-wideband systems that people are talking about now.

You use this very, very wide bandwidth, and you see how many signals you can pack into this very wide bandwidth using PAM. And what's the answer? There are 2 t0 w 0 real degrees of freedom. And a baseband bandwidth w 0. This is what we determine when we were dealing with PAM. It simply corresponded to the fact that the baseband bandwidth was 1 over 2 t 0. Therefore we got two signals coming in, each 1 over w 0. So this was the number of real degrees of freedom we have.

Now, the thing we want to do is to take this humongous bandwidth. And to say, what happens if instead of using this wide bandwidth system, we use a lot of little narrow bands. So we make narrow bands send QAM in the narrow bands, where in fact we're packing signals into the real part and the imaginary part, which is what we decided we ought to do. And how do these two systems compare.

Well, if we take a very wide bandwidth w0, with lots of little bandwidths, w, up at various passbands, all stacked on top of each other. Like here's zero. Here's w0 which is up at five gigahertz or whatever you want to make it. We put lots of little passbands in here, each of width w. And the question is how many different frequency intervals do we get? How many different signals can we multiplex

together here by using different frequency bands? Which is exactly the way people who use QAM normally transmit data.

Well, the answer is, you get w0 over w different bands here. So how many degrees of freedom do you have? Well, we said we had 2 t 0 w real degrees of freedom with each QAM system. We have w0 over w different QAM systems all stacked on top of each other. So, when you take w equals w0 over m you wind up with 2 t0 w, 2 t0 w 0 real degrees of freedom, overall using QAM. Exactly the same thing if you use PAM.

This corresponds to these orthonormal series we were thinking about, where we were thinking in terms of taking segmenting time, and then using a Fourier series within each segment of time. And looking at the number of different coefficients that we got. And we then looked at these sinc weighted expansions of the same type. We got the same number of degrees of freedom. This is simply saying that PAM and QAM use every one of those degrees of freedom. And there aren't any more degrees of freedom available. If you're stuck with using some very large time interval t0 and some very large frequency interval, w0, you're stuck with that many real degrees of freedom, and real waveform. There's nothing else available, and PAM and QAM are perfectly efficient as far as their user spectrum is concerned. They both do the same thing.

How about single sideband? That does the same thing too. I mean, with single sideband, you're sending real signals but you cut off half the bandwidth at passband. So, again, you get the same number of signals for unit time and per unit bandwidth. And most systems, except double sideband quadrature carrier, which is just obviously wasting bandwidth, all do the same thing. So, it's not hard to use all of these available degrees of freedom. And the only thing that makes these degrees of freedom arguments complicated is, there's no way you can find the pulse, which is perfectly limited to 1 over w in time and 1 over w in bandwidth. If you make it limited in time, it spills out in bandwidth. If you make it limited in bandwidth, it spills out in time. So we always have to fudge a little bit as far as seeing how these things go together.

Let's talk about one sort of added practical detail that you always need in any QAM system. In fact, you always need it in a PAM system, also. And you need to do it somehow or other.

There's this question of, how do you make sure that the receiver is working on the same clock as the transmitter is working on? How do you make sure that when we talk about a cosine wave at the receiver, the phase of that cosine wave is, quote, the same as the phase at the transmit?

So, if the transmitted waveform at the upper passband -- at positive frequencies is u of t times e to the 2 pi i f c t. This is this is one of the reasons why you want to think in terms of complex notation. If you try to analyze QAM phase lock using cosines and sines -- I mean, you're welcome to do it, but it's just a terrible mess. If you look at it in terms of complex signals, it's a very easy problem. But it's most easy when you look at it in terms of just what's going on at positive frequencies.

So the thing we're doing here is, when we modulate, we're taking u of t. We multiply by e to the 2 pi i f c t. What happens to any arbitrary phase that we have in the transmitter? Well, nothing happens to it. We just define time in such a way that it's not there.

At the receiver, we're going to be multiplying again by e to the minus 2 pi i f c t. There's going to be some phase in there because the receiver's clock and the transmitter's clock are not the same.

Now, the thing which is complicated about this is that at the receiver there's some propagation delay from what there is at the transmitter. So what we're really saying here is that when the waveform u of t gets to the receiver -- in other words, after we demodulate, what do we get? That's the practical question to ask. I mean you have an arbitrary sinusoid, complex sinusoid at the transmitter. You have an arbitrary complex sinusoid at the receiver. When you multiply by e to the 2 pi i f c t, and then at the receiver you demodulate by multiplying e to the minus 2 pi i f c t, there's some added phase in there just because you're multiplying by some sinusoid in an arbitrary phase.

What do you wind up with when you get done with that? You get the received waveform, u of t, times e to the i phi. Now, suppose you're sending analog data. What happens here? Well, with analog data, you're absolutely up a creek without an oar. There's nothing you can do. But with digital data there is something you can do. And I think this diagram makes it clear, if you can see the little arrows on it.

What you're transmitting is QAM signals out of a QAM constellation. So the signals -- so you're either transmitting this, or this, or this, or this, and so forth. Now, at the receiver we haven't analyzed noise yet. We're going to start doing that later today. But if you visualize noise being added up at passband and then coming down to baseband again, we can visualize what's happening as the noise just adds something to each one of these points.

I mean, assuming that we know exactly what the timing is for the time being. So that we sample at exactly the right time at the receiver. And that's a separate issue. But let's assume that we do know what the timing is. We do sample at the correct time. And the only question we're trying to find out now is, how do we choose this phase right on this sinusoid that we're using at the demodulator. Because at the demodulator, there's no way to know whether phi is zero or whether phi is something else.

But after we got all done doing this, if we sample at the right time, what we're going to receive, if there's some small error in phase, is that this diagram is all rotated around a little bit. Any time you send this point, what you receive is going to be a point up here. If you send this point it's going to be a point here. If phi is negative, everything will go the other wa.y How do I know whether it's positive phi or negative phi? I don't. I just do the same thing either place. Everybody I know who worries about phase locked loops ignores the sign and simply makes sure that they're using the same sign at the transmitter and the receiver. If they don't, when they build it, the thing goes out of lock immediately. And they try switching the sign, and if it works then, they say uh-huh, I now have the right sign. I mean, it's like these factors of two. You always put them in after you're done.

Anyway, I think I've done it right here. So the received waveform is now going to be u of t times e to the i phi. It will have a little bit of noise added to it. So what we're getting, then, if you look at a scope, or whatever people use as scope these days. And people have done this for so many years that they call this an eye pattern. And what they see, after looking at many, many received signals, is little blobs around each of these points due to noise. And they also see the set of points rotated around a little bit. And if the noise a small, and if the phase error is small, you can actually see this diagram rotated around sitting at some orientation. And the question is, if you know that's what's happening, what do you do about it?

Well, if you have this diagram rotated around a little bit, you want to change the phase of your demodulating carrier. So the thing you do is, you measure the errors that you're making when you go from these received signals. And you make decisions on them. You then look at the phase of these errors. If all the phases are positive, then you know that you have to change the phase of this carrier one way. And if all the phases are negative, you change it the other way.

Now, why does this work? It works because these clocks are very stable clocks. They wander in phase slowly. I mean, you can't lock them perfectly. And since they wander in phase slowly. And since you're receiving data quickly, you can average this carrier recovery over many, many data symbols. Which is what people do. So, you average it over many symbol. And you gradually change the phase as the phase is getting off.

How much trouble does that cause in trying to actually receive the data? Hardly any. Because if the phase changes are slow enough, then you average over a long enough interval, the actual phase errors are going to be extraordinarily small. So it's the noise errors that look big and the phase errors that look very small. When you average over a long period of time, the noise is sort of independent from one period to another. So the noise averages out. And the thing you see, then, is the phase errors. So, so long as the phase is slow and the noise is quick, you first decode the data. After you decode the data, you average over what the phase errors are. And then you go and you change the clock. And people spend their lives designing how

to make these phase lock loops. And there's a lot of interesting technology there.

But the point is, the idea is almost trivially simple. And this is what it is. Since the phase is changing slowly, you can change the phase slowly. You can keep the phase error very, very small. And therefore, the data recovery due to the noise works almost as well as if you had no phase error at all.

How do you recover from frequency errors? Well, if I can recover from phase errors, I'm recovering from frequency errors also. Because I had this clock which is running around. And I'm slaving the clock at the receiver to the clock at the transmitter. So, so long as I have very little frequency error to start with, simply by keeping this phase right, I can also keep the carrier frequency right. Because the phase is just going around at the speed it's supposed to be going around. So this will also serve as a frequency lock as well as a phase lock. When you're trying to do both together, you build the phase lock loop a little different, obviously.

What's the problem with this? You look this diagram, and you say, OK, suppose there's some huge error where for a long period of time, things get very noisy. What's going to happen?

Well, if you can't decode the data for a long period of time, suddenly these phase errors build up. The channel stops being noisy. And what could have happened is that your phase could be off by pi over 2. Which means what you see, what you should be seeing is this. What you're actually seeing is this. You can't tell the difference. In other words, you decode every symbol wrong. Because you're making an error of pi over 2 in every one of them. You decode this as this, you decode this as this, this as this, and so forth. So every point is wrong.

What will you do to recover from this? I mean, suppose you were designing a system where in fact you had no feedback to the transmitter at all. You have feedback from the transmitter, you just, obviously every once in a while you stop and you resynchronize. You don't have any feedback, what do you do? Yeah?

AUDIENCE: [INAUDIBLE]

PROFESSOR: Differential phase shift key, yes. I mean, this isn't quite phase shift key, but it's the same idea. In other words, the thing that you do is, you don't -- usually when the data is coming in, you don't map it directly into a signal point. The thing that you do is map the data into a phase change from one period of time to the next. I mean, you map it into an amplitude. And then you map it into a change of phase in this set of points. Or change of phase within this set of points. Or a change of phase within the outer set of points. And then if, due to some disaster, you get off by pi over 2, it won't hurt you. Because the changes are still the same, except for the errors that you make when this process starts.

So in fact, a lot of these things that people have been doing for years, you would think of almost immediately. The only thing you have to be careful about, when you invent new things to build new systems, is you have to have somebody to check whether somebody else has a patent on it. So, you invent something and then you check whether somebody else patented it. So that's what this slide says. Bingo.

It's time to go on to talk about additive noise and random processes. How many of you have seen random processes before? How many of you have seen it in a graduate context as opposed to an undergraduate context? A smaller number. Many of you, if you got your degrees outside of the US, if you've studied this as an undergraduate, you probably know what students here have learned in their various graduate courses. So you might know a lot more about it.

What we're going to do here is I'll try to teach you just enough about random processes that you can figure out what's going on with the kinds of noise that we're looking at this term. Which involves two kinds of noise. One, something called additive noise, which is what we're going to be dealing with now. And the other is much later, when we start studying wireless systems, we have to deal with a different kind of noise. People sometimes call it multiplicative noise, but really what it is, is these channels fade in various ways. And the fading is a random phenomena. And therefore we have to learn how to deal with that in some kind of probabilistic way also.

So the thing we're going to do is we're going to visualize what gets received as what gets transmitted plus noise. Now, at this point, I've said OK, we know how to deal with compensating for phase errors. So we'll assume that away, it's our usual layered approach. What kind of other assumptions are we making here? When I say the received signal is the transmitted signal plus the noise.

Well, I mean, I could look at this and say, this doesn't tell me anything. This is simply defining the noise. There's the difference between what I receive and what I transmit. So when I say this is additive noise, I'm not saying anything other than the definition of the noise, is this difference. But when we look at this, we're really going to be interested in trying to analyze what this process is. And we're going to be interested in analyzing what this is as a process also. Here, I'm just looking at sample functions. And what I'd like to be able to do is say this sample function, namely, what's going into the channel, is one of a large set of possibilities. If it weren't one of a large set of possibilities, there would be no sense at all to transmitting it, because the receiver would know what it was going to be. And you wouldn't have to transmit it, it would just print it out at the output. So, this is some kind of random process which is determined by the binary data which is coming into the encoder and the signals it gets mapped into, and so forth. This reviewing of some kind of noise waveform.

What we're going to assume is that in fact these two random phenomena are independent of each other. And when I write this, you would believe that if nobody told you to think about it. Because it just looks like that's what it's saying. Here's some noise phenomena that's added to the signal phenomenon. If in fact I told you, well, this noise here happens to be twice the input then you'd say, foul. What I'm really doing is calling something noise which is really just a scaling of the input. Which is really a more trivial problem. So we don't want to do that.

Another thing I could try to do, and people tried to do for a long time in the communication field, was to say, well, studying random processes is too complicated. Let me just say there's a range of possibilities, possible things for what I transmit. There's a range of possible waveforms for the noise. And I want to make

a system that works for both these ranges of different things. And it just turns out that that doesn't work. It works pretty well for the transmitter. It doesn't work at all for the noise. Because sometimes the noise is what it should be. And sometimes the noise is abnormally large. And when you're trying to decide what kind of coding devices you want to use, what kind of modulation you want to use, and all these other questions, you really need to know something about the probabilistic structure here.

So we're going to view this as a sample function of a random process. And that means we're going to have to say what a random process is. And so far, the only thing we're going to say is, when we talk about it as a random process, we'll call it capital N of t instead of little n of t. And the idea there is that for every time, for every instant of time, when you talking about random processes, you usually call instants of time epochs. And you call them epochs because you're using time for too many different things. And calling it an epoch helps you keep straight what you're talking about.

So for each epoch t, n of t is a random variable. And this little n of t is just some sample value of that random variable.

So we're going to assume that we have some probabilistic description of this very large collection of random variables. And the thing which makes this a little bit tricky, mathematically is that we have an uncountably infinite number of random of variables to worry about here.

x sub t is known at the transmitter but unknown at the receiver. And what we're eventually concerned about is, how does the receiver figure out what was transmitted. So we'd better view x of t as a sample function of a random process. x of t, from the receiver's viewpoint. Namely, the transmitter knows what it is. This is this continual problem we run into with probability. When you define a probability model for something, it always depends on whose viewpoint you're taking. I mean you could view my voice waveform as a random process. And in a sense it is. As far as I'm concerned it's almost a random process, because I don't always know what

I'm saying. But as far as you're concerned, it's much more of a random process than it is from my viewpoint. If you understand the English language, which all of you do, fortunately, it's one kind of random process. If you don't understand that, it's just some noise waveform where you need a much more complicated model to deal with it. So everything we have to do here is based on these models of these random processes, which are based on reasonable assumptions of what we're trying to do with the model.

When we do this in terms of random processes, we have the input random process, and we have the noise random process. And if we add up two random processes, it doesn't take much imagination to figure out that what we get is another random process. It's random both because of this and because of this. Now, we implicitly are assuming a whole bunch of things when we write things out this way. We're not explicitly assuming them, but what we want to do is explain the fact that we're implicitly assuming them and then make it explicit.

One of those things that we're doing here is, we're avoiding all attenuation in what's been transmitted. We've been doing that all along. We've been talking about received waveforms as being equal to the transmitted waveform. Which means that we're assuming that the receiver knows what the attenuation is. And if it knows what the attenuation is, there's no point in putting in this scale factor everywhere. So we might as well look at what's being transmitted as being the same as what's being received, subject to this attenuation, which is just some known factor which we're not going to consider.

There's also this question of delay. And we're assuming that the delay is known perfectly. So that assumption is built into here. y of t is x of t without delay, plus z of t.

And now, after studying these modulation things, we can add a few more things which are implicitly put into here. One of them is that we know what the received phase is. So that in fact we don't have any phase error on all of this. So we're saying, let's get rid of that also. So we have no timing error, no phase error. All of

these errors are disappearing because what we want to focus on is just the errors that come from this additive noise. Whatever the additive noise is. But we want to assume that this additive noise is independent of this process.

So we're saying, when we know all of these things about -- or when the receiver knows all of these things about what's being transmitted, except for the actual noise waveform, how do we deal with the problem?

We implicitly mean that z of t is independent of x of t, because otherwise it would be just very misleading to define n z of t as the difference between what you receive and what you transmit. So we're going to make that assumption explicit here. These are standing assumptions until we start to study wireless systems. And another standing assumption which we'll make is the kind of assumption we've been making in this course all along. Which some people like, usually theoreticians. And some people often don't like, namely practitioners. But which if both the theoreticians and practitioners understood better what the game was, I think they would agree to doing things this way.

And what we're going to do here is, we're not going to think of what happens when people make measurements of this noise and then try to create a stochastic model of it from all of these measurements. What we're going to do is more in the light of what Shannon did then when he invented information theory. Which is to say, let's try to understand the very simplest situations first. If we can understand those very simplest situations namely, we will invent whatever noise we want to invent. To start to understand what noise does to communication. And we will look at the simplest processes we can. Which seem to make any sense at all of us. And then after we understand a few of those, then we will say, OK, what happens if there's some particular phenomenon going on in this channel.

And when you look at what practitioners do, what in fact they do is, everything they do is based on various rules of thumb. Where do these rules of thumb come from? They come from the theoretical papers that were written 50 years before, if they're very out of date practitioners. 25 years before if they're relatively up to date

practitioners. Or one year before if they're really reading all the literature as it comes out. But rules of thumb usually aren't based on what's happening as far as papers are concerned now. It's usually based on some composite of what people have understood over a large number of years.

So in fact, the way you understand the subject is to create these toy models. And the toy models are what we're going to be creating now. Understand the toy models, and then move from the toy models to trying to understand other things. This is the process we'll be going through when we look at this additive noise channel, which doesn't make much sense for wireless channels, as it turns out. But it's part of what you deal with on a wireless channel. And then, on a wireless channel, we'll say, well, are these other effects concerned also? And we'll try to deal with these other effects as an addition to the effects that we know how to deal width. We've already done that as far as phase noise is concerned. Namely, we've already said that since phase effects occur slowly, we can in fact get rid of them almost entirely. And therefore we don't have to worry about them here. So, in fact we have done what we've said that we believe ought to be done. Namely, since we now know how to get rid of the phase noise, we're now looking at additive noise and saying, how can we get rid of that.

Actually, if you look at the argument we went through with phase noise, we were really assuming that we knew how to deal with channel noise as part of that. And in fact when you get done knowing how to deal with channel noise, you can go back to that phase locked loop argument. And you'll see a little more clearly, or a little more precisely, what's going on there.

When we started talking about waveforms, if you remember back to when we were talking about compression, we really faked this issue of random processes. If you remember the thing we did, everything was random while we were talking about taking discrete sources and coding for them. When we then looked at how do you do quantization on sequences of real numbers or sequences of complex numbers, we again took a probabilistic model. And we determined how to do quantization in terms of that probabilistic model. And in terms of that probabilistic model, we figured

out what ought to be done.

And then, if you remember, when we went to the problem of going from waveforms to sequences, and our pattern was, take a waveform. Turn it into a sequence. Quantize the sequence, then encode the digital sequence. We cheated. Because we didn't talk at all about random processes there. And the thing we did instead is, we simply converted source waveforms to sequences and said that only the probabilistic description of the sequence is relevant.

And, you know, that's actually true. Because so long as you decide that what you're going to do is go from waveforms to sequences, and in fact you have decided on which particular orthonormal expansion you're going to use, then everything from there on is determined just by what you know about the probabilistic description of that sequence. And everything else is irrelevant. I mean, each waveform maps into a sequence. When you're going back, sequences map into waveforms, at least in this L2 sense, which is what determines energy difference. And energy differences usually are criterion for whether we've done a good job or not. So, in fact, we could avoid the problem that way simply by looking at the sequences.

And it makes some sense to do that because we're looking at mean square error. And that was why we looked at mean square error.

But now we can't do that any more. Because now, in fact, the noise is occurring on the waveform itself. So we can't just say, let's turn this into a sequence and see what the noise on the waveform does to the sequence. Because, in fact, that won't work at this point.

So random process z of t is a collection of random variables, one for each t and r. So, in other words, this is really a collection of an uncountably infinite number of random variables. And for each epoch t, the random variable z of t is a function. Capital Z of t and omega. Omega is the sample point we have in this sample space. Where does the sample space come from? We're imagining it. We don't know any way to construct a sample space very well. But we know if we study probability theory that we need a sample space to talk about probabilities. So we're going to

imagine different sample spaces and see what the consequences of those are. I mean, that's the point of this general point of view that we have. That we start out looking at simple models and go from there to complex models. Part of the model is the sample space that we have.

So for each sample point that we're dealing with, the collection -- I should have written this differently. Let me change this a little bit to make it a little clearer. z of t omega for t and r is a sample function z of t; t and r.

So if I look at this, as a function of two variables. One is time and one is this random collection of things we can't see. If I hold omega fixed, this thing becomes a function of time. And in fact it's a sample function, it's the sample function corresponding to that particular element in the sample space. And don't try to ask what the sample space is. The sample space -- I mean, the sample point here -- really specifies what this sample function is. It specifies what the input sample function is. It specifies what everything else you're interested in is. So it's just an abstract entity to say, if we're constructing probabilities we need something to construct them on. So that's the sample space that we can construct these probabilities on.

The random processes -- Oh. OK. We want to look at two different things here. Here we're looking at a fixed sample point, which means that this process for a fixed sample point corresponds to a sample function. If instead I look a fixed time, a fixed epoch, t and r, and I say what does this correspond to as I look at z of t and omega over all omega in this big messy thing, fix t. What do I have there? I have a random variable. If I look at a fixed t, and I'm looking over the whole sample space, that's what you mean by a random variable. If I look at -- if I hold omega fixed and look over the time axis, what I have is a sample function.

I hope that makes it a little clearer what the game we're playing is. Because this random thing, the random process that we're dealing with, we can best visualize it as a function of two things, both time and sample point. The sample point in this big sample space is what controls everything. It tells you what every random variable in the world is. And what the relationship of every random variable in the world is.

So, a random process is defined, then, by some rule which establishes a joint density for all finite values of k, all particular sets of time, and all particular arguments here. What am I trying to do here? I'm trying to admit right from the outset that if I try to define this process for an uncountably infinite number of times, I'm going to be dead in the water. So I had better be happy with the idea of, at least in principle I would like to be able to find out what the joint probability density is for any finite set of points in time. And if I can find that, I will say I have defined this random process. And I'll see what I can do with it.

So we need rules that let us find this kind of probability density. That's the whole objective here. Because I don't know how to generate a random process that does more than that, that talks about everything all at once. You can read mathematics texts to try to do this, but I don't know any way of getting from that kind of general mathematics down to anything that we could make sense about in terms of models of reality.

The favorite way we're going to generate this kind of rule is to think of the random process as being a sum of random variables multiplied by orthonormal functions. In other words, we will have sample values in time, z of t, which are equal to z sub i phi sub i of t. And for each sample function in time, we will then have, for that particular time, as I vary over capital Omega, I'm going to be varying over the values of z sub i. These are just fixed functions. And what that's going to give me is a set of random variables here.

So in a way we're doing very much the same thing as we did with functions. With functions we said, when we're dealing with these the equivalence classes, things get very, very complicated. The way we're going to visualize a function is in terms of an orthonormal expansion. And with orthonormal expansions, if two functions are equivalent, they both have the same orthonormal expension. So now we're saying the same thing when we're dealing with this more general thing, which is stochastic processes.

This is really what the signal space viewpoint of communication is. It's the viewpoint

that you view these random processes in terms of orthonormal expansions where the coefficients are random variables. And if I deal with a finite number of random variables here, everything is very simple. Because you know how to add random variables, right? I mean, at least in principle, you all know how to add random variables. You've been doing that any time you take a probability class. A quarter of the exercises you deal with are adding random variables together and finding various things about the sum of those random variables. So that's the same thing we're going to be doing here. The only thing is, we can do this for any real value of t, just by adding up this fixed collection of random variables. And you add it up at a different arguments here because as t changes, phi sub i of t changes.

And this is sort of what we did when we were talking about source waveforms. It's very much the same idea that we were looking at. Because with source waveforms, when we turned the waveform into a sequence we were turning the waveform into a sequence by in fact using an orthonormal expansion. And then what was happening is that the coefficients in the orthonormal expansion, we were using to actually represent the random process. So we're doing the same thing here. The same argument.

So, somehow we have to figure out what these joint densities are. So, as I said before, when we don't know what to do, we make a simple assumption. And the simple assumption we're going to make is that these random variables at different points in time, of noise, we're going to assume that they're Gaussian. And a normal Gaussian random variable; this, I hope, is familiar to you, has the density 1 over the square root of 2 pi, e to the minus n squared over 2. So it's just density that has this nice bell-shaped curve. It's an extraordinarily smooth thing. It has a mean of 0. It has a variance of 1. How many of you could prove that the variance of this is 1? Well, in a while you'll all be able to prove it. Because there are easy ways of doing it. Other than looking at a table of integrals, which we'll do.

An arbitrary Gaussian random variable is just where you take this normal variable. These normalized Gaussian random variables are also traditionally called normal variables. I mean, they have such a central position in all of probability. And when

somebody talks about a normal random variable -- I mean, there's sort of the sense that random variable should be normal. That's the typical thing. It's what should happen. And any time you don't have normal random variables, you're dealing with something bizarre and strange. And that's carrying it a little too far, but in fact, when we're looking at noise, that's not a bad idea.

If you shift this random variable by a mean z bar, and you scale it by -- excuse me, scale it by sigma. Then the density of the result becomes 1 over the square root of 2 pi sigma squared, e to the minus z minus z bar over 2 sigma squared. In other words, the shifting just shifts the whole thing by z bar. So this density peaks up around z bar, rather than peaking up around 0, which makes sense. And if you scale it by sigma, then you need a sigma in here, too. When you have this probability density, you're going to be scaling it this way. Scaling it that way is what we mean by scaling it. If you scale it this way to make sigma squared larger, the variance larger, this thing still has to integrate to 1. So you have to pop it down a little bit also. Which is why this sigma squared appears here in the denominator. So anytime we scale a random variable, it's got to be scaled this way. And it's also got to be scaled down. So that's why these Gaussian random variables look this way.

We're going to be dealing with these Gaussian random variables so much that we'll refer to this random variable as being normal with mean z bar and sigma variance sigma squared. And if we don't say anything other than it's normal, we mean it's normal with mean 0 and variance 1. Which is the nicest case of all. It's this nice bell-shaped curve which everything should correspond to.

They tend to be good models of things for a number of reasons. One of them is the central limit theorem. And the central limit theorem, I'm not going to state it precisely here. I'm certainly not going to prove it. One of the nastiest things to prove in probability theory is the central limit theorem. It's an absolute bear. I mean, it seems so simple. And you try to prove it, and it's ugly.

But, anyway, the central limit theorem says that if you add up a large number of Gaussian random variables, of independent Gaussian random variables, and you

scale the sum down to make it variance one, you have to scale it down. Then when you get all done adding them all up, what you wind up with is a Gaussian random variable.

This is saying something a little more precise than the law of large numbers. The law of large numbers says you add up a bunch of independent or almost independent random variables. And you scale them down. And you get a mean, which is something. This is saying something more. It's saying you add them all up, and you scale them down in the right way. And you get not only the mean, now, but you also get the shape of the density around the mean. Which is this Gaussian shape, if you add a lot of them.

It has a bunch of extremal properties. In a sense, it's the most random of all random variables. Not going to talk about that, but every once in a while, one of these things will come up. When you start doing an optimization in probability theory, over all possible random variables of a given mean and variance, you often find that the answer that you come up with either when you maximize or when you minimize is Gaussian. If you get this answer when you maximize, then what you get when you minimize is usually a random variable, which is zero or something. And vice versa. So that either the minimum or the maximum of a problem makes sense. And the one that makes sense, the answer is usually Gaussian. At least, it's Gaussian for the problems for which it's Gaussian. But it's Gaussian for an awful lot of problems.

It's very easy to manipulate analytically. Now, you look at that density, and you say, my God, that can't be easy to manipulate analytically. But in fact, after you get used to two or three very small tricks, in fact you'll find out that it's very easy to manipulate analytically. It's one of the easiest random variables to work with. Far easier than a binary random variable. If we add up a bunch of binary random variables, and what do you get? You get a binomial random variable. And binomial random variables, when you add up enough of them, they get uglier and uglier. Except they start to look Gaussian.

These Gaussian random variables, if you add up a bunch of them, and what do you

get? You get a Gaussian random variable. Yes?

AUDIENCE: Central limit theorem applies to random variables or Gaussian random variables?

PROFESSOR: Central limit theorem applies to random variables.

AUDIENCE: [INAUDIBLE]

PROFESSOR: Oh, I'm sorry. I shouldn't have. No. I said, when you add a bunch of Gaussian random variables, you get exactly a Gaussian random -- when you add up a bunch of jointly Gaussian random variables, and we'll find out what that means in a while, you get exactly a Gaussian random variable. When you add up a bunch of random variables which are independent, or which are close enough to independent, and which have several restrictions on them and you scale them the right way. What you get as you add more and more of them, tends toward a Gaussian distribution. And I don't want to state all of those conditions.

I mean, we don't need all those conditions here. Because the only thing we're relying on is when we look at noise, we're looking at a collection of a very large number of phenomena. And because we're looking at a very large number of phenomena, we can sort of model it as being an addition of a lot of little random variables. And because of that, in an enormous number of situations, when you look at these things, what you get is fairly close to a Gaussian shape.

And it's a model that everybody uses for noise. If you read papers in the communication field, or in the control field, or in a bunch of other fields, and people start talking about what happens when noise enters the picture, sometimes they will talk about their model for the noise process. Sometimes they won't. But if they don't talk about what noise model they're using, it's a very safe bet that what they're using is a Gaussian process model.

In other words, this is the kind of model people use because it's a model that makes a certain amount of sense.

Now, there's one other thing you ought to be aware of here. It's something that

we've talked about a little bit in other contexts. When we're looking at a noise waveform that you have to deal with in a modem or something like that, this modem only experiences one noise waveform in its life. So, what do we mean by saying that it's random? Well, it's an act of faith, in a sense. But it's not so much. Because if you're designing communication equipment, and you're cookie cuttering cellphones or something like that, or something else, what you're doing is designing a piece of equipment which is supposed to work in an enormous number of different places. So that each one you're manufacturing is experiencing a different waveform. So that as far as the designing the device is concerned, what you're interested in is not that one waveform that one of these things experiences, but in fact the ensemble of waveforms that all of them experience.

Now, when you look at the ensemble of waveforms that all of these things experience, over all of this large variety of different contexts; noise here, noise there, noise there, suddenly the model of a large collection of very small things makes even better sense. Because in each context, in each place, you have some forms of noise. When you look at this over all the different contexts that you're looking at, then suddenly you have many, many more things that you, in a sense, are just adding up if you want to make a model here.

So we're going to use this model for a while. We will find out that it leads us to lots of nice things. And so forth.

So, we said that we'd be happy if we could deal with at least studying a random process over some finite set of times. So it makes sense to look at a finite set of random variables. So we're going to look at a k-tuple of random variables. And we're going to call that -- what do you think we're going to call a k-tuple of random variables? We're going to call it a random vector. In fact, when you look at random vectors, you can describe them as being elements of a vector space. And we're not going to do that because most of you are already sufficiently confused about doing functions as vectors, and adding on the idea of a random process as a vector would be very unkind. And we're not going to do it because we don't need to do it. We're simply calling these things vectors because we want to use the notation of vector

space, which will be convenient here.

So, if we have a collection of random variables n 1 to n k, and they're all IID, independent identically distributed. They're all normal with mean zero and variance one, the joint density of them is what? Well, you know what the joint density is. If they're independent and they're all the same, you just multiply their densities together. So it's 1 over 2 pi with a k over 2. e to the minus n 1 squared minus n 2 squared, and so forth. Divided by 2. That's why this density form is a particularly convenient thing. You have something in the exponent. When you have independent random variables, you multiply densities. Which means you add what's in the exponent. Which means everything is very nice. So you wind up adding up all these sample values for each of the random variables. With our notation looking at norm squared, this is just the norm squared for this sequence of sample values. And look at the sequence of samples values is a actual vector, n 1 up to n sub k. So we wind up width the norm of the vector n squared divided by 2.

If you draw a picture of that, you have spherical symmetry. Every set of sample vectors which have the same energy in them has the same probability density. If you draw it in two dimensions, you get -- it looks like a target where the biggest probability density is in here, and the probability density goes down as you move out. If you look at it in three dimensions, if that same thing's looked at in a sphere. And if you look at it in four dimensions, well, it's the same thing, but I can't draw it, I can't imagine it. But it's still a spherical symmetry.

In the one minute that I have remaining, I'm going to -- I might derive a Gaussian density. I'm going to call a bunch of random variables zero-mean jointly Gaussian if they're all derived by linear combinations on some set of normal IID Gaussian in random variables. In other words, if each of them is a linear combination of these IID random variables, I will call the collection z 1 up to z k a jointly Gaussian random variable.

In other words, jointly Gaussian does not mean that z 1 is Gaussian, z 2 is Gaussian, up to z k being Gaussian. It means much more than that. It means that

all of them are linear combinations of some underlying set of independent random variables. In the homework that was passed out this time, you will look at two different ways of generating a pair of random variables which are each Gaussian, but which are not jointly Gaussian. Which you can't view in this way. So this is important.

Jointly Gaussian makes sense physically because when we look at random variables, we say it makes sense to look at Gaussian random variables because of a collection of a lot of noise variables. If you look at two different random variables, each of them is going to be a collection of different small noise variables. And they're each going to be some weighted sum over a different set of noise variables. So it's reasonable to expect each of them to be expressible in this way. I guess I'm not going to derive what I wanted to derive. I'll do it next time.

It's easier to think of this in matrix form. In other words, I'll take the vector of sample values, z 1 up to z sub k. Each one of them is a linear combination of these normal random variables, n 1 up to n sub n. So I can do it as z equal to matrix a times n. This is a vector of real numbers. This is a vector of real numbers, this is a matrix. And if I make this a square matrix, than what we're doing is mapping each of the unit vectors, each single noise vector, into the i'th column of a.

In other words, if I have a noise vector where n 1 is equal to 0, n 2 is equal to 0, n sub i is equal to 1 and everything else is equal to 0, what that generates in terms of this random vector z is just the vector a sub 1. So. What I'm going to do when I come back to this next time is, if I take a little cube in the n space, and I map it into what goes on in the z space, this unit vector here is going to map into a vector here. The unit vector here is going to map into a vector here. So cubes map into parallelograms.

Next time, the thing we will say is, if you look at the determinant of a matrix, the determinant of a matrix, the most fundamental definition of it, is that it's the volume of a parallelepiped. In fact, it's the volume of this parallelepiped. So we'll do that next time.